

09-29-00

A

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE  
**UTILITY PATENT APPLICATION TRANSMITTAL**

ASSISTANT COMMISSIONER OF PATENTS  
Box Patent Application  
Washington, D.C. 20231

Attorney Docket No. 21891.03200

Prior Application Number:

Art Unit:

Examiner:

PTO Customer Number:

Sir:

Transmitted herewith for filing is a utility patent application of

Sherry Solden, 4377 Miller Avenue, Palo Alto, CA 94306

Edwin A. Harcourt, 4 Gemini Lane, Townsend, MA 01469

William W. La Rue, Jr., 15296 Sherwood Road, Leawood, KS 66224

Douglas D. Dunlop, 11912 Hunting Ridge Court, Potomac, MD 20854

Christopher Hoover, 741 Pollard Road #4, Los Gatos, CA 95032

Qizhang Chao, 2796 Louis Road, Palo Alto, CA 94303

Poonam Agrawal, 1063 Morse Avenue, #20-106, Sunnyvale, CA 94089

Aaron Beverly, 15245 Upper Ellen Road, Los Gatos, CA 95030

Massimiliano L. Chiodo, 10 Hillcrest Road, Berkeley, CA 94705

Neeti K. Bhatnagar, 4301 Rennaissance Drive #127, San Jose, CA 95134

Soumya Desai, Manhar Mahal, 4 Bakul Bagan Row, Flat #4C, Calcutta 700025, India

Hungming Chou, 6822 Corte Munras, Pleasanton, CA 94566

Michael D. Sholes, 1318 Edgewood Road, Redwood City, CA 94062

Ian Dennison, 555 River Oaks Parkway, San Jose, CA 95134

Sanjay Chakravarty, M36, Ramkrishna Vihar 29, I P Extension, Delhi 110092, India

Eamonn O'Brien-Strain, 555 River Oaks Parkway, San Jose, CA 95134

Luciano Lavagno, 1571 Scenic Avenue, Berkeley, CA 94708

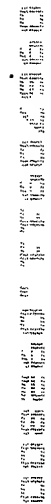
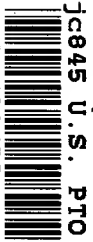
for: METHOD AND SYSTEM FOR PERFORMANCE LEVEL MODELING AND SIMULATION OF  
ELECTRONIC SYSTEMS HAVING BOTH HARDWARE AND SOFTWARE ELEMENTS

Enclosed are the following:

1. ☒ Specification, abstract and claims of 31 pages.
2. ☒ 9 sheets of ☐ formal ☒ informal drawings ☐ No drawings.
3. ☒ Declaration (original or copy) by the named inventor(s) - unsigned
4. ☐ Preliminary Amendment
5. ☐ Information Disclosure Statement
6. ☐ Other:

**NOTE the following:**

7. ☐ Applicant is a small entity. Unsigned Small Entity Statement enclosed – 50% Filing Fee Reduction  
(if applicable)
8. ☐ The prior application is assigned to



9. This application is a:

☐ Continuation ☐ Divisional ☐ Continuation-in-Part (CIP) of Prior Application

Filed:

10. ☐ Priority of the following application(s) is (are) claimed under 35 U.S.C. 119:

| Serial No. | Date Filed | Country | Certified Copy of Priority Doc. Filed |      |
|------------|------------|---------|---------------------------------------|------|
|            |            |         | USSN or PCT#                          | Date |
|            |            |         |                                       |      |
|            |            |         |                                       |      |

11. ☐ An Extension of Time is filed concurrently herewith for the parent application.

12. ☐ Cancel claims \_\_\_\_\_ prior to calculation of the filing fee.

**FILING FEE:**calculated below (after accounting for any preliminary amendment or claims cancellations if noted above):

|                                     |                                 |    |                                 |          |
|-------------------------------------|---------------------------------|----|---------------------------------|----------|
| <input checked="" type="checkbox"/> | Total Claims                    | 28 | Total Claims Subject to Fees: 8 | \$144.00 |
| <input checked="" type="checkbox"/> | Independent Claims              | 3  | Total Claims Subject to Fees: 0 | \$       |
| <input type="checkbox"/>            | Multiple Dependent Claims       |    |                                 | \$       |
| <input checked="" type="checkbox"/> | Basic Filing Fee                |    |                                 | \$690.00 |
| <input type="checkbox"/>            | Extension Fees                  |    |                                 | \$       |
|                                     | Sub-Total                       |    |                                 | \$834.00 |
| <input type="checkbox"/>            | Less Small Entity Fee Reduction |    |                                 | \$       |
| <input type="checkbox"/>            | Assignment Recordal Fees        |    |                                 | \$       |
|                                     | Total Fees                      |    |                                 | \$834.00 |

☒ Check(s) no. 472210 in the amount of \$834.00 is enclosed (must at least cover the basic fee). If no check or an insufficient check is enclosed and a fee is due herewith, the Commissioner is authorized to charge any fee or additional fee due in connection herewith to Deposit Account No. 03-3821, referencing Attorney Docket No. 21891.03200. A duplicate of this sheet is enclosed.

[illegible]

Adrian H. Z...

Rebecca M. Glits  
Signature of Mailing Individual

**METHOD AND SYSTEM FOR PERFORMANCE LEVEL MODELING  
AND SIMULATION OF ELECTRONIC SYSTEMS HAVING BOTH  
HARDWARE AND SOFTWARE ELEMENTS**

Inventors: Sherry Solden

Edwin A. Harcourt, William W. LaRue, Douglas D. Dunlop, Christopher Hoover,  
Qizhang Chao, Poonam Agrawal, Aaron Beverly, Massimiliano L. Chiodo,  
Neeti K. Bhatnagar, Soumya Desai, Hungming Chou, Michael D. Sholes, Ian  
Dennison, Sanjay Chakravarty, Eamonn O'Brien Strain, Luciano Lavagno

BACKGROUND

FIELD OF THE INVENTION

The present invention relates generally to designing and evaluating performance level models of electronic systems and amongst other things to a method and system for creating and simulating models of electronic systems at the system level.

BACKGROUND OF THE INVENTION

In the traditional electronic system design process, behavior and architecture specification are followed by hardware and software design. The opportunity to consider tradeoffs in function and architecture performance occurs too late in the design flow for any changes to be made in either a timely or cost-effective manner.

The design process of the products for these systems is subject to a number of constraints. A first constraint is that they must be implemented in silicon or another hardware platform for power, performance and cost reasons. A second constraint is that these products implement systems conceived by a highly specialized system team thinking in terms of executable concurrent programming paradigms which, today, are not well understood by hardware designers. In fact,

Patent Application

Docket Number: 21891.03200

in most systems the partitioning of functions between hardware and software is based upon designer's past experience and is not subject to any analysis. Then, the partitioned specifications are translated into a specific hardware description language (HDL) such as Very High Speed Integrated Circuit (VHSIC) Hardware  
5 Description Language (VHDL) or Verilog for the hardware components and a software description language such as C or assembler for the software components. Although the hardware and software have tight interaction, both hardware and software are designed separately. Only after the system is assembled is the software and hardware run together. As a consequence, the design can be far from optimal or even erroneous, making a redesign cycle mandatory.

This gap between system design and implementation is rapidly becoming the most problematic bottleneck in the design process of such products and systems. At the same time the conditions of the marketplace have created a need to quickly design products because of time to market requirements. Therefore, as the design time grows the time allowed for design in the business cycle continuously decreases. A major approach to shortening the design time is by attempting to implement hardware/software co-design procedures, so that the hardware and software of a system can be concurrently designed in order to speed up the design  
20 process. However, efficient co-design methodologies and approaches have not been easy to conceive or implement. One of the reasons for this is that the methodologies for hardware design and software design have their own approaches that are difficult to mesh.

Several approaches have been attempted to create a true co-design  
25 methodology. One known approach is a co-verification approach, where designed hardware and software are verified together utilizing a co-verification simulator. A problem with this approach is that all the hardware must be built and designed to the cycle accurate level at this point, and therefore any problems that arise during co-verification, if they can be addressed by redesigning the hardware, are difficult

to implement as the hardware has already been designed and implemented.  
Similarly, the software for the system needs to be compiled prior to co-verification.

Another approach to co-design is by instruction set (ISS) co-simulation. This approach concurrently simulates the instruction sets for a processor and related components by utilizing an HDL description of the system and a model of the instruction set. While this can lead to an effective co-simulation, this approach still utilizes a model of the hardware that is functionally complete making it difficult to replace or substantially redesign any hardware components. This approach also requires a complicated processor model which is difficult to create and thus provides a high barrier to exploration of various processors. This approach too results in hardware over-design, with the related higher costs, power consumption, and equipment sizes in systems.

Therefore, there is a need for a hardware/software co-design methodology that allows for simulation at a level where the hardware is not yet completely designed, to allow simple redesign of the hardware components.

Another issue in the design of digital systems is the ability to reuse components that were used in previously designed systems. The design process for a digital system must allow for the reuse of components, and therefore support a reusable design methodology. The problem in reusing previously designed components lies in the fixed communication protocols they use, which necessitates protocol conversions when different components with different protocols have to be interfaced. In fact, it has been noted that more than half of all designs are reused in building future systems.

Today, the selection of a protocol is done while designing the component: functional and communication behavior are intrinsically mixed. However, a good selection of the protocol is possible only when all components involved in the communication are known. Therefore, a design environment for digital systems should permit a component to be initially described in purely functional terms. Later, when the component is (re)used in a system, the design environment must

allow to plug in the most appropriate communication behavior. This approach is in contrast with current hardware design practices, where communication and functional behavior are mixed.

The ability to reuse components requires component modularity. In modular designs, the complete system functionality is split into communicating components of manageable complexity. The advantage of this approach is that the components can be reused and that the system is easier to adapt and maintain.

Additionally, the following requirements should be considered for a hardware/software system design environment. (1) modularity is essential to reduce complexity; (2) multiple description languages should be accommodated to allow each system component to be described within the most appropriate paradigm; (3) the design environment must be able to model the heterogeneous conceptual specification, the resulting heterogeneous architecture and all refinement steps in between; and (4) off-the-shelf components and the associated design environments should be modeled.

## SUMMARY OF THE INVENTION

The present invention is directed toward a method and system of generating and evaluating performance level models of electronic systems having both hardware and software components. The present invention provides a simplified method for allowing system designers to specify the functional components of their systems and then utilizing the specified components to map and evaluate the mapped model of the system.

In one embodiment, the present invention is directed toward a method of modeling an electronic system having both hardware and software elements. The method comprises generating a plurality of behaviors, associating each of the behaviors of the plurality of behaviors to an appropriate architectural component, and generating communication patterns between the architectural components that require communication between them in order to perform the behaviors.

In another embodiment, the present invention is directed toward a system for creating a behavioral model of an electronic system having hardware and software components. The system comprises a plurality of architectural components, each corresponding to a component capable of being implemented as part of the electronic system, and means for generating communication patterns between the architectural components that require communication between them in order to perform user specified behaviors.

In a further embodiment, the present invention is directed toward a performance level model of an electronic system having hardware and software elements. The model comprises an input function that provides input information, a first service that represents a function to be performed by a first architectural component of the electronic system, a second service that represents a function to be performed by a second architectural component of the electronic system, at least one application programming interface that facilitates communication between the first service and the second service, and an output function that

receives output information of the performance level model of the electronic system.

It is an object of an aspect of the present invention to provide system level simulation functionality to allow a designer to work at high levels of abstraction and with larger, more complex circuits and systems.

It is a further object of an aspect of the present invention to provide a technique to easily implement and test several different architectural designs for compliance with the desired operational requirements of a designed system.

It is another object of an aspect of the invention to raise the level of design validation to a system level.

It is an additional object of an aspect of the invention to provide a more standardized design environment, thereby alleviating the need for cross-training between different design platforms and allowing resources to be directed more towards design and implementation.

It is yet another object of an aspect of the invention to provide an intuitive and interactive technique for designing complex digital systems.

It is yet a further object of an aspect of the present invention to provide a technique for allowing high level iterative refinement of a complex digital system design.

It is therefore an advantage of the invention to provide a method of modeling an electronic system having both hardware and software elements, the method comprising capturing a plurality of behaviors that correspond to operations performed by the system being modeled, capturing a plurality of hardware and software architectural components the plurality being contained within an architectural platform, mapping each of the captured behaviors of the plurality of behaviors to a selected architectural component to perform the behavior, recognizing and capturing communication patterns among the architectural components that require communication among them in order to perform the

behaviors, and mapping each instance of communication between behaviors to an instance of the captured pattern.

Another advantage of the invention is to provide a system for creating a behavioral model of an electronic system having hardware and software components, the system comprising a plurality of architectural components, each of the architectural components corresponding to a component capable of being implemented as part of the electronic system, and means for generating communication patterns between the architectural components that require communication between them in order to perform user specified behaviors, each communication pattern including communication between any intervening architectural components needed to communicate between architectural components carrying out the behaviors.

Still another advantage of the invention is to provide a performance level model of the communications between behaviors of an electronic system having hardware and software components, the model comprising an application programming interface on one behavior that provides data to be transferred to one or more destination behaviors, a first service that implements the application interface which models the performance of the communication protocol, the service being among a plurality of services supported by the pattern to which the behavior communication is mapped, one or more application programming interfaces by the first service to model the performance of the architecture platform, the application interfaces being among a plurality of service declarations supported by the symbol of the architectural component to which the first behavior is mapped, a supported service declaration on the symbol of the architecture component that is implemented by a service definition, the service definition being among a plurality of service definitions specified by the performance model of the architecture component, a second application interface that represents a function to be performed by a second architectural component topologically connected to

the first component of the electronic system, the second service being one a plurality of second services each corresponding to a function capable of being performed by the second architectural component, and an input application interface on the destination behavior that receives output information of the performance level model of the electronic system, thereby completing the communication from source behavior to destination behavior.

10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100  
101  
102  
103  
104  
105  
106  
107  
108  
109  
110  
111  
112  
113  
114  
115  
116  
117  
118  
119  
120  
121  
122  
123  
124  
125  
126  
127  
128  
129  
130  
131  
132  
133  
134  
135  
136  
137  
138  
139  
140  
141  
142  
143  
144  
145  
146  
147  
148  
149  
150  
151  
152  
153  
154  
155  
156  
157  
158  
159  
160  
161  
162  
163  
164  
165  
166  
167  
168  
169  
170  
171  
172  
173  
174  
175  
176  
177  
178  
179  
180  
181  
182  
183  
184  
185  
186  
187  
188  
189  
190  
191  
192  
193  
194  
195  
196  
197  
198  
199  
200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211  
212  
213  
214  
215  
216  
217  
218  
219  
220  
221  
222  
223  
224  
225  
226  
227  
228  
229  
230  
231  
232  
233  
234  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244  
245  
246  
247  
248  
249  
250  
251  
252  
253  
254  
255  
256  
257  
258  
259  
260  
261  
262  
263  
264  
265  
266  
267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323  
324  
325  
326  
327  
328  
329  
330  
331  
332  
333  
334  
335  
336  
337  
338  
339  
340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447  
448  
449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537  
538  
539  
540  
541  
542  
543  
544  
545  
546  
547  
548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
600  
601  
602  
603  
604  
605  
606  
607  
608  
609  
610  
611  
612  
613  
614  
615  
616  
617  
618  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
740  
741  
742  
743  
744  
745  
746  
747  
748  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
840  
841  
842  
843  
844  
845  
846  
847  
848  
849  
850  
851  
852  
853  
854  
855  
856  
857  
858  
859  
860  
861  
862  
863  
864  
865  
866  
867  
868  
869  
870  
871  
872  
873  
874  
875  
876  
877  
878  
879  
880  
881  
882  
883  
884  
885  
886  
887  
888  
889  
890  
891  
892  
893  
894  
895  
896  
897  
898  
899  
900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
970  
971  
972  
973  
974  
975  
976  
977  
978  
979  
980  
981  
982  
983  
984  
985  
986  
987  
988  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
999  
1000

## BRIEF DESCRIPTION OF THE DRAWINGS

The aforementioned advantages of the present invention as well as additional advantages thereof will be more clearly understood hereinafter as a result of a detailed description of a preferred embodiment of the invention when  
5 taken in conjunction with the following drawings.

Fig. 1 is a flow chart of a design process according to a presently preferred embodiment of the present invention;

Fig. 2 is a flow chart of a method of generating a behavioral and performance model according to a presently preferred embodiment of the present  
10 invention;

Fig. 3 is a diagram of a mapping procedure according to a presently preferred embodiment of the present invention;

Fig. 4 is a diagram of an exemplary architectural component according to a presently preferred embodiment of the present invention;

Fig. 5 is a diagram of a communication pattern according to a presently preferred embodiment of the present invention;

Fig. 6 is a diagram of a communication pattern with corresponding pattern and architectural component services according to preferred embodiment of the present invention; and  
15

Fig. 7 is a diagram showing the interaction between services communicating across a bus according to a presently preferred embodiment of the present  
20 invention; and

Fig. 8 is a diagram showing the interaction between services communicating across a bus bridge according to a presently preferred embodiment of the present  
25 invention; and

Fig. 9 is a diagram showing the interaction between performance models of services according to a presently preferred embodiment of the present invention.

While the invention is susceptible to various modifications and alternative forms, specific embodiments thereof have been shown by way of example in the

Patent Application

Docket Number: 21891.03200

5

## DESCRIPTION OF THE DRAWINGS

The present invention allows the user to clearly differentiate between a behavior model, which identifies what the system does, and an architecture model, which identifies the components that implement the system. This clear  
5 differentiation between system function and architecture allows system designers to simulate the performance effects of a behavior running on a number of different architectures early in the design cycle.

In designing a system having both hardware and software elements there are several important factors that should be analyzed. Such things include the  
10 execution delay behavior of the system processor(s), the overhead and functionality of the scheduler of the operating system, the execution delay of any system application specific integrated circuit (ASIC) or on-chip custom hardware, the communication paths and timing between hardware and software components, delays with respect to bus and memory access, and timing for instruction and data  
15 fetching. In addition, contention problems between resources can also be obtained from system level simulations and therefore dealt with early in the design process.

Referring to Fig. 1, a user inputs the desired system behavior , step 10. The user then inputs the architecture components that implement the system, step 15. The behaviors that the user has input are then mapped to the appropriate  
20 architectural components, step 20. Once the behaviors are mapped to the architectural components the performance of the mapped system can be simulated, step 25. The performance simulation determines, for a particular mapped design, whether the timing of the system meets the user's requirements. If not, the user can map portion of the behavioral blocks to different architectural blocks, possibly  
25 causing their implementation to move between hardware and software. The design may be a system-on-a-chip with embedded processors, memories, and custom hardware, or it may be composed of discrete processors, memories, and Systems On A Chip (SOCs). When the design is at the fully refined level and its performance meets the system requirements, the user can export it as a software

and hardware implementation, step 30. The hardware design will then be ready for HDL simulation, floor planning, and logic synthesis, step 35. The software models will be ready for linking to an RTOS, step 4. The hardware and software components can then both be separately verified, steps 45 and 50. Additionally the hardware and software components of the exported design will be able to be verified together in a co-verification tool, step 55.

Referring to Fig. 2, a plurality of behaviors are generated by a user, step 100. The behaviors can be input by the user as C++ objects, selected from a database of behaviors, composed hierarchically in block diagram, be input into dialog boxes, or otherwise generated. The behaviors describe the desired functions of the system that the designer wants the system to be able to perform.

A plurality of architecture components are generated by a user, step 110. The architecture components can be input, selected from a database of architectures, composed hierarchically in an architecture diagram, be input into dialog boxes or otherwise generated. As used herein, architectural components refers to a model of an architectural element, which is a realizable component of an actual system. Examples of architectural elements include busses, CPU's, Real Time Operating Systems (RTOS), Schedulers, ASICs, etc. The architectural components include a number of services that each relate to a specific function that is performed by the architectural element. For instance, a RTOS can be modeled by a scheduler application, a standard C Library, etc.

The behaviors are then partitioned into hardware and software by mapping the behavior instances to the appropriate architecture component (e.g. RTOS or ASIC), step 120. The user can now run a performance analysis of their system which models the impact of this particular partition, step 130. The user iterates the partitioning step until the system meets the performance requirements.

Once the partitioning is complete, the user can refine the mapped design by choosing communication patterns for the plurality of nets in behavior design, step 140. The communication patterns include the timing, speed and protocols that are



architectural component. In this way each architectural component is broken into multiple functions, increasing modularity and promoting reusability of services between architecture components. For example, there are only a handful of scheduling algorithms which are used by all the RTOSes. A service for round robin scheduling could be defined once and used in many RTOS components. The services and their function can be determined based upon the architectural component specifications provided by the vendor that provides the architectural component that is being modeled.

It is presently preferred that each service is defined by a service declaration and one or more service definitions. A service declaration declares in a C++ header file one or more C++ functions. The service definition provides the body of the C++ function declared by the header file. The advantage of this approach is that C++ is a standard, object-oriented language which supports separation of function declaration and definition.

The service declarations for each architecture component are selected from a library or other database that is either provided by the manufacturer of the component represented by the architectural component or an industry-driven standard library or created by the system user. This approach promotes industry standards for modeling architecture components that can be mixed and matched; which in turn promotes exploration of architecture designs by the system designer.

The service definitions for each architectural component are selected from a library or other database that is either provided by the manufacturer of the component represented by the architectural component or can be created by the system user. The advantage of this approach is that components from different vendors can work together as long as they implement services with matching APIs. In addition, if the definition is provided by the IP vendor, the system designer doesn't have to understand all the complicated details because they are encapsulated within the service. Another advantage of this approach is that many service definitions can be written to model different levels of performance accuracy

while retaining the same interface. The system designer can easily and gradually move to higher levels of accuracy in their performance analysis and the design becomes more refined.

To follow the RTOS example, it may support the StandardCLibrary service.

5 The service declares the function prototype for "memcpy" and "memset". The implementation of the service would define the functionality and performance impact of these two functions.

The function prototypes for memcpy and memset are as follows:

10 virtual vccAddress\* memcpy(vccAddress\* s1, const vccAddress\* s2, size\_t n,  
vccInstance\*) = 0;  
virtual vccAddress\* memset(vccAddress\* s, int c, size\_t n, vccInstance\*) = 0;

15 For still additional guidance, the presently preferred C++ code for memcpy and memset is as follows:

/\*\*\*\*\*\*

Name: memcpy

Description: Copies n bytes from memory address s2 to s1. This

implementation assumes that addresses lies on memory

20 word boundaries. No actual data is designed to be

stored or read. Memory transactions are generated for

performance reasons only.

Return: Always returns s1.

\*\*\*\*\*/

25 vccAddress\* CPP\_MODEL\_IMPLEMENTATION::memcpy(vccAddress\* to, const  
vccAddress\* from,

size\_t n, vccInstance\* inst)

{

if (n == 0)

30 return to;

```

Init();
unsigned reqTrans = n / bytesPerWord_;
reqTrans = ((reqTrans==0) ? 1 : reqTrans);
unsigned remTrans = reqTrans;
5 theBegin:
  if (remTrans == 0) goto theEnd;
  {
    memAccess.reference(*from, bytesPerWord_, rwRead,inst,true);
    memAccess.reference(*to, bytesPerWord_, rwWrite,inst,true);
10  }
    remTrans--;
    goto theBegin;
theEnd:
    return to;
15  }

```

```

/*****

```

Name:      memset

Description: Sets the first n bytes of memory at address s to the value  
of c (converted to an unsigned char). Memory transactions  
are generated for performance reasons only.

Return:     Always returns s;

```

*****/

```

```

25 vccAddress* CPP_MODEL_IMPLEMENTATION::memset(vccAddress* s, int c, size_t

```

n,

```

        vccInstance* inst)

```

```

{
  if (n == 0)
30  return s;

```

```

Init();
vccAddress* to = s;
unsigned remTrans = n;
theBegin:
5   if (remTrans == 0) goto theEnd;
    {
        memAccess.reference(*to, bytesPerWord_, rwWrite,inst,true);
    }
    remTrans--;
10  goto theBegin;
theEnd:
    return s;
}

```

Referring to Fig. 5, an example of a communication pattern is depicted between two behavioral blocks 300 and 310 that is mapped to the appropriate architectural components, in this case RTOS 320. A communication arc between the two behavioral blocks 300 and 310 is mapped to the appropriate pattern, in this case Semaphore pattern 330. The Semaphore pattern 330 implements a communication protocol composed of a pair of pattern services 340 ("sender") and 350 ("receiver"), one for each end of the communication pattern 330. The "sender" service models the locking of the mutex, writing the data, unlocking the mutex and sending a trigger to the destination behavior. The "receiver" models locking the mutex, reading the data and unlocking the mutex.

It is presently preferred that the pattern services 340 and 350 are not directly mapped to architectural components. Instead the mapping of each pattern service 340 and 350 is implied by the mapping of the behavior block to an architectural block. That is, since the pattern services 340 and 350 implements or refines part of the behavior, it uses resources or services of the architectural



component 450 is a software component that controls the CPU architectural component 470.

To start the communication pattern, a post function 490 is called from within the behavior component 400. The post function provides the input to the sender pattern service 500 of the communication pattern 420. The register mapped sender models the transfer of the data from the software to a register on the ASIC (to which the behavior component 410 is mapped). In this example, behavior component 410 is mapped to ASIC 460. The sender service implements the data transfer using the "memcpy" function provided by the RTOS "Standard C Library" service. Each service declares that it uses a set of service declarations and the service definitions of each service must be found on connected architecture components. The pattern sender service can use services provided by the architecture component to which the source behavior (400) is mapped. In this case, the Standard C Library is found on the RTOS 450. The RTOS provides the implementation of the standard C library service, which in this example uses a memory access interface to write data to the register on the ASIC. The memory access service definition is found on the CPU 470 because the RTOS can use services provided by the processor to which it is assigned. The CPU memory access service 540 uses the bus adapter model 550 which is found on the port of the CPU. The bus adapter model 550 uses the bus arbiter service on the BUS architecture component 480, and once granted bus ownership it uses the slave adapter service 560 on the port of the ASIC architectural component 460, which is the service that models the bus communication of the ASIC. The slave adapter service 560 uses the slave service on the ASIC 460 to store the data in the local register. From the other side of the pattern, the register mapped receiver 580 implements the value and enable functions 590, using the slave service to retrieve the data written, thus completing the pattern protocol.

The advantage of this approach is that the communication patterns are based on the architecture topology with each of the components supplying

modular, reusable services. This approach provides a process for architectural exploration at the system level.

Searching for pattern services on connected architecture components works well when there is a single communication between components. A bus inherently violates this principle because it is a medium for multiple pairs of communications. Referring to Fig. 7, behavior 600 is mapped to an RTOS 620 and behavior 610 is mapped to ASIC 660. The communication pattern uses services on the RTOS, CPU, bus and the ASIC 660. A generic bus adapter 635 on the CPU needs to differentiate that this message should be sent to the slave adapter 665 on ASIC2, while other messages might be sent to other slave devices (ASICs) on the bus. The distinction can not rely on the search for a matching service definition, because each slave device will support the same slave bus adapter. It is presently preferred that a symbolic address be sent along with each bus transaction so that it can be directed to the correct slave adapter service. In this approach, the symbolic address consists of the architecture instance name and an offset. Each slave registers its slave adapter service by instance name with the bus registry. When a pattern sends a message it must provide an address of the message which can be quickly looked up in the registry to find the appropriate slave adapter. The pattern sender and receiver services must declare the message sequence required to complete the pattern protocol as path specifications, e.g.:

*Path : <pathName> <sourceArchInst> <destArchInst> <offsetparam> <dataType>.*

Sometimes the source and destination of each message is the architecture components to which the behaviors are mapped. For example, in the register mapped pattern, the sender sends data from the RTOS to ASIC2 and these are the architecture components to which beh1 and beh2 are mapped. In other scenarios, a third architecture component may participate in the message sequence. For example in a shared memory register mapped pattern, the data is first written to a RAM component. It is presently preferred, that the sourceArchInst and destArchInst can reference the keywords vccArchOfSrcBehav or

vccArchOfDestBehav or can reference a parameter of the pattern service which can then be further exported to the pattern instance and specified by the end-user during the mapping process. In the latter scenario, the end-user selects the memory participant from the set of memories in the architecture diagram for each usage of a shared memory pattern. The dataType specifies whether the message sends the data on the net or just a trigger. This information is used to determine the performance impact on the bus because it accurately accounts for the size of the data being transferred. The advantage of this approach is that patterns can be reused across different architectures.

Referring to Fig 8, the behaviors are mapped to the RTOS 720 and ASIC2, 780. In this example, the message must be sent from RTOS across Bus1 740 through BusBridge1 760 across Bus2 770 and finally to ASIC2. The bus bridge facilitates conversion of bus transactions from Bus1 to Bus2. It is presently preferred that the bus bridge registers all slave services on Bus2 with Bus1 registry. The usage of architecture instances (which is unique across the architecture diagram) ensures that the slave services on Bus2 will not conflict with slave services on Bus1. As a result, the symbolic address will reference instance name "ASIC2" and the Bus1 registry will find the bus bridge slave adapter. It is also presently preferred that the bus bridge will have slave adapter 765 on one port and a bus master adapter 775 on a second port. The slave adapter 765 connected to bus1 will receive the bus request and then the bus master adapter on the second port will resend the request across bus2 for which ASIC2 will now handle the transfer. Since bus bridges may have more than 2 ports, it is also presently preferred that each slave port must identify via a port attribute the master port to which it resends the data. The advantage of this approach is that an architecture diagram can be easily extended or reorganized around busses and bridges to optimize the overall system performance.

A protocol registry which is a database of patterns that can be searched using some key is presently preferred to be utilized so that once the behavior

components are mapped all that is required by the system designer is to select from a list of patterns. A description of some of the presently preferred patterns is described herein for the purposes of example only and is not intended to be limiting in any way.

The patterns of communication are arranged in basic groups, depending on whether the implementation chosen for the sender/receiver is an ASIC (HW), a separate SW task (SW inter-task), or within the same SW task (SW intra-task), and whether the communication only deals with transmitting the presence of the event (Trigger) or data is also involved. For each group there is a choice of possible patterns:

- 1) HW – HW: (a) Direct Connect, (b) Register Mapped, (c) Shared Memory:
- 2) HW - HW Trigger: (a) Direct Connect, (b) Register Mapped
- 3) HW – SW: (a) Interrupt Register Mapped, (b) Interrupt Shared Memory, (c) Polling Register Mapped, (d) Polling Shared Memory
- 4) HW - SW Trigger: (a) Interrupt, (b) Polling Register Mapped, (c) Polling Shared Memory
- 5) SW – HW: (a) Register Mapped, (b) Shared Memory
- 6) SW - HW Trigger: (a) Register Mapped
- 7) SW - SW Inter-task: (a) Unprotected, (b) Semaphore Protected, (c) Uninterruptable Protected
- 8) SW - SW Inter-task Trigger: (a) Unprotected
- 9) SW - SW Intra-task: (a) Unprotected
- 10) SW – SW Intra-task Trigger: (a) Unprotected.
- 11) SW->Memory (a) SWDirectMemoryAccess (b) SWDMAAccess
- 12) HW->Memory (a) HWDirectMemoryAccess (b) HWDMAAccess

- 13) SW->Timer (a) SWVirtualTimer  
14) HW->Timer (a) ASICInternalTimer

The advantage of grouping the patterns is that the communication arcs can be categorized into each of the groups based on the mappings of the behavior components as well as the size of the data type on the arc. Once the user selects a default pattern for each of the basic groups the unmapped arcs can be automatically assigned to these defaults saving time in the mapping process. The user can always explicitly map or change the mapping to a different choice but the defaults should be applicable a large percentage of the time.

Referring to Fig. 9, a first behavior block 800 sends an event to a second behavioral block 810 using a communication pattern 820. The communication pattern 820 comprises services P1...Pn on the sender's side and services Q1...Qn on the receiver side. An output port 830, in the first behavior block 800, and an input port in second behavior block 810 begin and terminate the communication pattern 820. The control flows of the communication pattern 820 is as follows: the delay model of first behavior block 800 calls the behavioral block 800, which calls the function Post which is implemented by a function in P1. P1 calls some function of P2, P2 calls some function of P3 etc. At the end of the implementation of Post there is a call to an input change function that posts the event to the second behavioral block 810 with the required delay. This triggers, at a later time, the delay model of second behavioral block 810, which first calls the top receiver pattern service Q1 to set the value and event buffers in the instance of Q1, and then calls the appropriate service of the second behavioral block 810 which calls an enable function which in turn calls some function of Q1. Q1 calls some function of Q2, Q2 calls some function of Q3, etc. until Enabled() returns a result. The process may be repeated for any additional values and operations. It should be noted that the delay model as referred to herein is the delay of the architectural component performing the behavior, due to scheduling or buffering or the like.

It is presently preferred, as depicted in Fig. 9, that a communication pattern 820 is bound to a port, e.g. port 830 or 840, rather than to a net. In the case of a net with multiple fanouts, each destination port may be associated with a different communication pattern although it is presently preferred that nets from each architectural component use the same communication pattern for all fanouts. In a communication net with N fanouts each using a different communication pattern, the call to the post function in the behavioral block is going to be implemented as a sequence of N calls to the various implementations of the post function for each of the communication patterns.

This preferably requires an intermediate layer of code between the sender's behavioral block and the sender's top pattern service. That is, the implementation of the post function called by the behavior model should be a loop over the fanouts, and for each of them it will call the implementation of the post function provided by the service used by the pattern associated to the fanout. In the simulation, a C++ behavior block calls the implementation of the post function in the service directly. There is a code layer in the pattern that received the post from the models and then unravels it out to each of the patterns. This is performed inside a specific service which will receive the list of actual services, one per fanout, that it needs to call. This dispatcher service may be dropped when only one communication pattern is used on all fanouts of a net.

Referring again to Fig. 9, the data transmission from the post function 830 to the value function 840 in the simulation is inherently safe as it is performed atomically by a call to input changed function. The simulation of the finer details of the transfer only deals with the timing aspects, not the content.

While the embodiments, applications and advantages of the present inventions have been depicted and described, there are many more embodiments, applications and advantages possible without deviating from the spirit of the inventive concepts described and depicted herein. The invention should only be restricted in accordance with the spirit of the claims appended hereto and is not



## CLAIMS

What is claimed is:

1. A method of modeling an electronic system having both hardware and software elements, comprising:

5 capturing a plurality of behaviors that correspond to operations performed by the system being modeled;

capturing a plurality of hardware and software architectural components the plurality being contained within an architectural platform;

10 mapping each of the captured behaviors of the plurality of behaviors to a selected architectural component to perform the behavior;

15 recognizing and capturing communication patterns among the architectural components that require communication among them in order to perform the behaviors; and

20 mapping each instance of communication between behaviors to an instance of the captured pattern.

2. The method of Claim 1 wherein each architectural component comprises a plurality of services each service corresponding to a particular function of the architectural component.

3. The method of Claim 2 wherein each service comprises a declaration and a definition corresponding to the interface and body of the service, respectively.

4. The method of Claim 3 wherein each service definition uses other service declarations.

5. The method of Claim 1 wherein the step of capturing the architectural platform comprises selecting a performance model for each of the plurality of architectural components wherein the services of one architectural component uses other services that are topologically connected in the platform.

6. The method of Claim 1 further comprising a step of partitioning each behavior of the plurality of behaviors that are captured as either a hardware or software behavior.

7. The method of Claim 1 wherein said capturing communication pattern step comprises a plurality of services each service uses other service declarations that are supported by the architecture component to which the behavior is mapped.

8. The method of Claim 1 further comprising modifying at least one behavior of the plurality of behaviors or associating at least one behavior with a new appropriate architectural component while maintaining associations among the other behaviors with previously mapped architectural components.

9. The method of Claim 1 further comprising associating at least one behavior of the plurality of behaviors with a new architectural component and recognizing and capturing one or more new communication patterns while maintaining other previously captured communication patterns.

10. The method of Claim 1 further comprising changing the selection of one or more pattern mappings while maintaining other previously captured communication patterns.

11. The method of Claim 1 further comprising modifying at least one architecture component of the plurality of architecture components in the platform while maintaining associations amongst the other architecture components and patterns of communication to and from the modified component. This modification can be exchange of component for another component or a tweak to the parameter settings or a different selection of performance model.

12. The method of Claim 1 further comprising modifying the plurality of architecture components that are contained in an architecture while maintaining previously captured behavior mappings and communication patterns.

13. The method of Claim 1 further comprising at least one member of a group consisting of reuse of services in other architecture components and reuse of architecture components in other architecture platforms or reuse of architecture platforms in other electronic systems.

14. The method of Claim 1 further comprising reuse of patterns in other electronic systems with variations on the architecture platform.

15. The method of Claim 1 further comprising simulating operation of the electronic system when performing the plurality of behaviors utilizing the architecture services of the mapped behaviors and optionally the pattern services that were selected.

16. The method of Claim 15 wherein utilizing the architecture services can be successively refined where additional design decisions are made causing the performance model to become more accurate.

17. A system for creating a behavioral model of an electronic system having hardware and software components, comprising:

a plurality of architectural components, each of the architectural components corresponding to a component capable of being implemented as part of the electronic system; and

means for generating communication patterns between the architectural components that require communication between them in order to perform user specified behaviors, each communication pattern including communication between any intervening architectural components needed to communicate between architectural components carrying out the behaviors.

18. The system of Claim 17 wherein each architectural component comprises a plurality of services each service corresponding to a particular function of the architectural component.

19. The system of Claim 17 wherein the means for generating communication patterns between the architectural components comprises the means for selecting the appropriate services of the plurality of services necessary for carrying out communication between the architectural components.

25 20 The system of Claim 17 further comprising means for associating each behavior  
of a plurality of behaviors with an appropriate architectural component.

21. The system of Claim 17 wherein the means for associating each behavior of the plurality of behaviors comprises means for selecting an architectural

component model for the architectural component, the architectural component model being one of a group of predefined architectural component models.

22. The system of Claim 17 further comprising means for classifying each behavior of a plurality of behaviors to be performed by the electronic system as either a hardware or software behavior.

23. The system of Claim 17 further comprising means for associating at least one modified behavior with a new appropriate architectural component while maintaining associations of the other unmodified behaviors with previously mapped architectural components.

24. The system of Claim 17 further comprising means for associating at least one behavior of a plurality of behaviors with a new architectural component.

25. The system of Claim 24 further comprising means for generating one or more new communication patterns between the new architectural component and other architectural components while maintaining other previously generated communication patterns.

26. The system of Claim 17 further comprising a simulation application for simulating operation of the electronic system when performing a plurality of behaviors utilizing the communication patterns that were generated.

27. A performance level model of the communications between behaviors of an electronic system having hardware and software components, the model comprising:

an application programming interface for a first behavior that provides data to be transferred to one or more destination behaviors;

a first service that implements the application programming interface that models the performance of the communication protocol, the first service being among a plurality of services supported by the pattern to which the behavior communication is mapped;

one or more application programming interfaces used by the first service to model performance of the architecture platform, the application interfaces being

among a plurality of service declarations supported by the symbol of the architectural component to which the first behavior is mapped;

a supported service declaration on the symbol of the architecture component by a service definition, the service definition being among a plurality of service definitions specified by the performance model of the architecture component;

a second application interface that represents a function to be performed by a second architectural component topologically connected to the first component of the electronic system, the second service being one a plurality of second services each corresponding to a function capable of being performed by the second architectural component;

an input application interface on the destination behavior that receives output information of the performance level model of the electronic system, thereby completing a communication from source behavior to destination behavior.

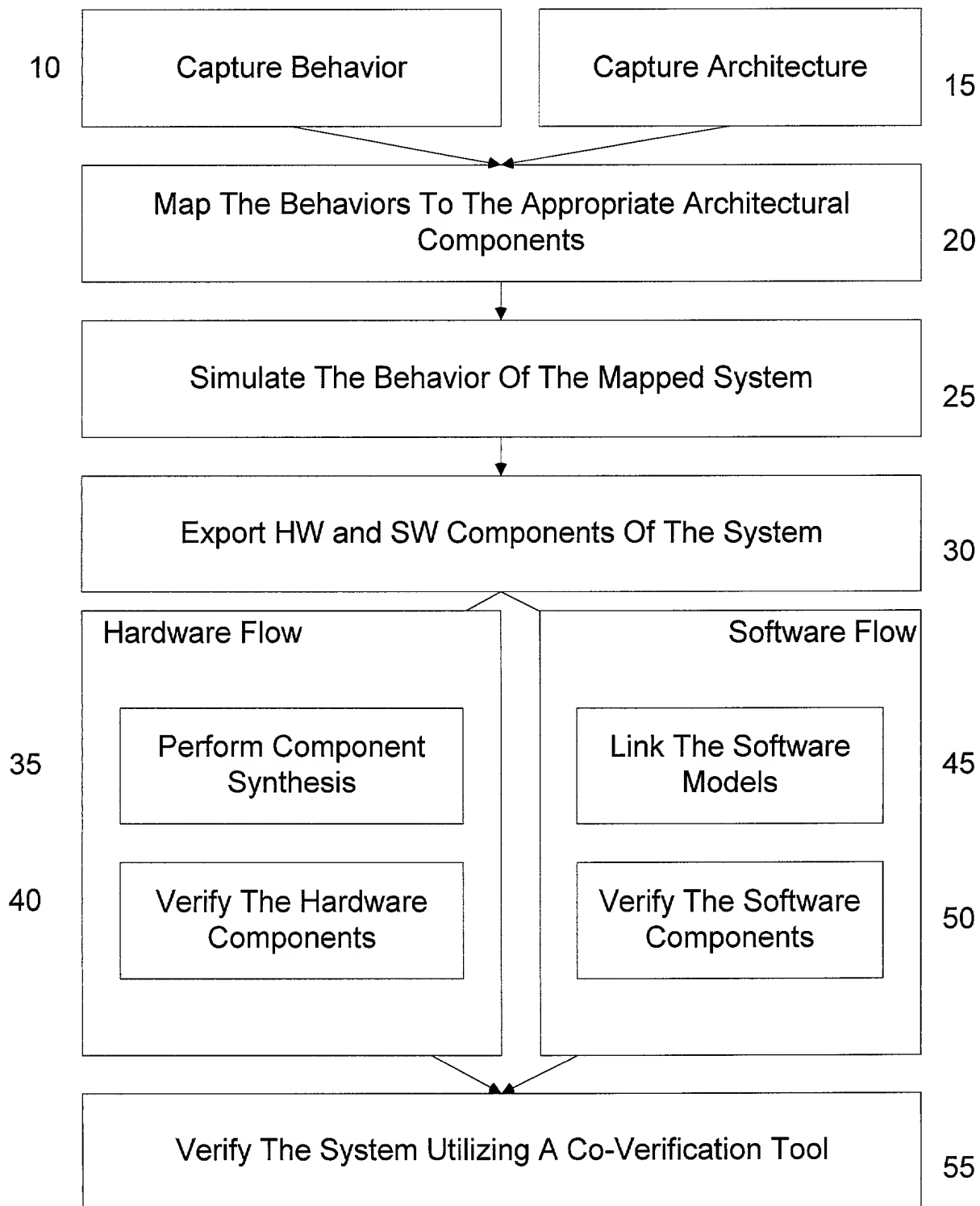
28. The performance level model of Claim 27 further comprising a third service that represents a function to be performed by a third architectural component of the electronic system, the third service being one a plurality of services each corresponding to a function to be performed by the third architectural component and wherein there is at least one application programming interface comprising a first application programming interface that is a model of the communication behavior between the first and second architectural components topologically connected and a second application programming interface that is a model of the communication behavior between the second and third topologically connected architectural components.

## ABSTRACT

A method and system for evaluating performance level models of electronic systems having both hardware and software components is provided. The system and method allow for the simplified implementation and testing of several different architectural designs for compliance with the desired operational requirement of a designed electronic system.

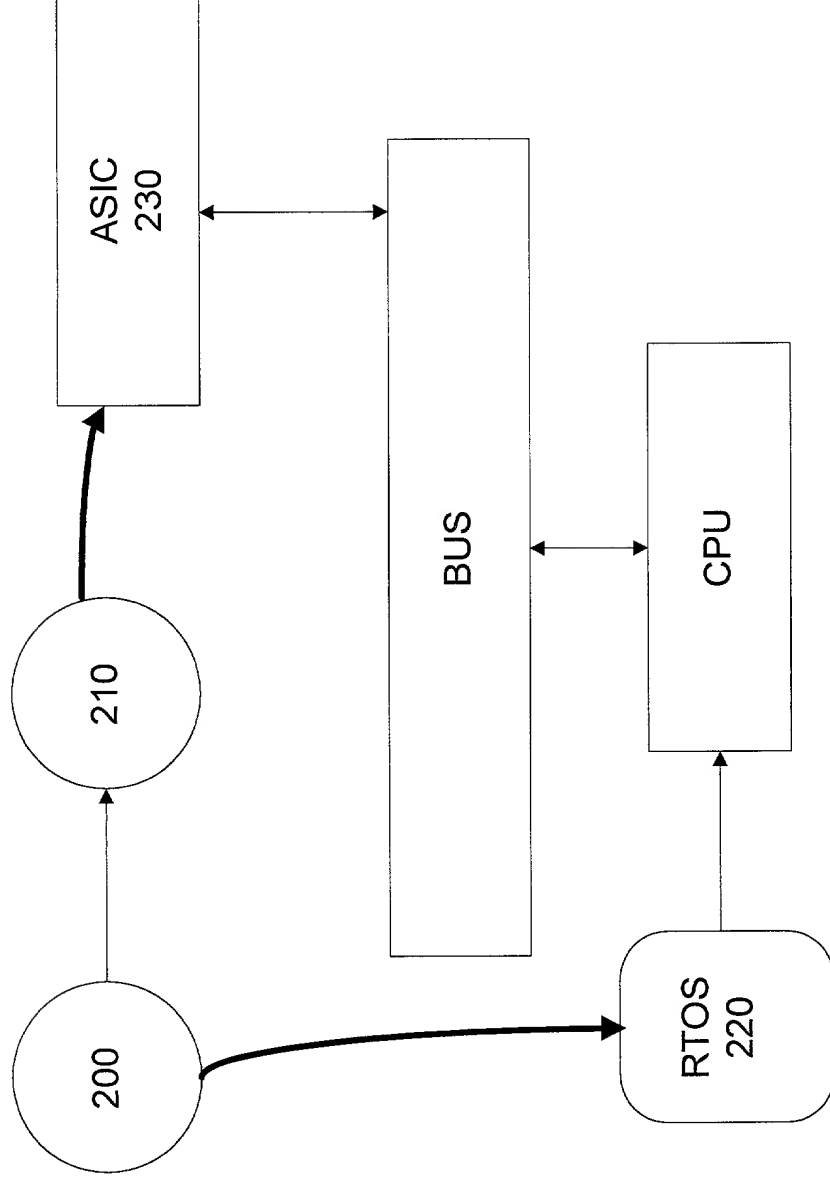
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100  
101  
102  
103  
104  
105  
106  
107  
108  
109  
110  
111  
112  
113  
114  
115  
116  
117  
118  
119  
120  
121  
122  
123  
124  
125  
126  
127  
128  
129  
130  
131  
132  
133  
134  
135  
136  
137  
138  
139  
140  
141  
142  
143  
144  
145  
146  
147  
148  
149  
150  
151  
152  
153  
154  
155  
156  
157  
158  
159  
160  
161  
162  
163  
164  
165  
166  
167  
168  
169  
170  
171  
172  
173  
174  
175  
176  
177  
178  
179  
180  
181  
182  
183  
184  
185  
186  
187  
188  
189  
190  
191  
192  
193  
194  
195  
196  
197  
198  
199  
200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211  
212  
213  
214  
215  
216  
217  
218  
219  
220  
221  
222  
223  
224  
225  
226  
227  
228  
229  
230  
231  
232  
233  
234  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244  
245  
246  
247  
248  
249  
250  
251  
252  
253  
254  
255  
256  
257  
258  
259  
260  
261  
262  
263  
264  
265  
266  
267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323  
324  
325  
326  
327  
328  
329  
330  
331  
332  
333  
334  
335  
336  
337  
338  
339  
340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447  
448  
449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537  
538  
539  
540  
541  
542  
543  
544  
545  
546  
547  
548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
600  
601  
602  
603  
604  
605  
606  
607  
608  
609  
610  
611  
612  
613  
614  
615  
616  
617  
618  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
740  
741  
742  
743  
744  
745  
746  
747  
748  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
840  
841  
842  
843  
844  
845  
846  
847  
848  
849  
850  
851  
852  
853  
854  
855  
856  
857  
858  
859  
860  
861  
862  
863  
864  
865  
866  
867  
868  
869  
870  
871  
872  
873  
874  
875  
876  
877  
878  
879  
880  
881  
882  
883  
884  
885  
886  
887  
888  
889  
890  
891  
892  
893  
894  
895  
896  
897  
898  
899  
900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
970  
971  
972  
973  
974  
975  
976  
977  
978  
979  
980  
981  
982  
983  
984  
985  
986  
987  
988  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
999  
1000

12226617.1

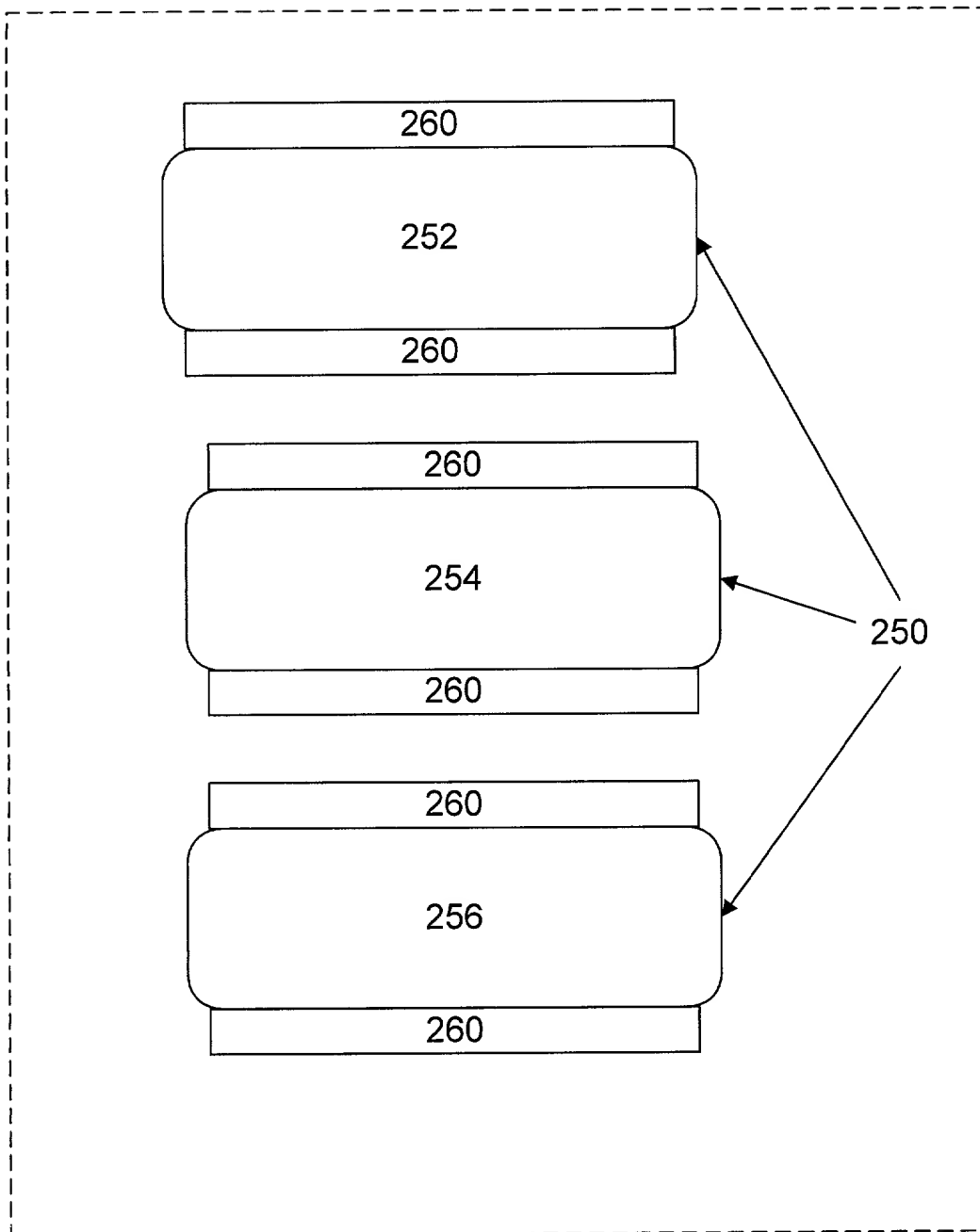


**Fig. 1**

**Fig. 2**

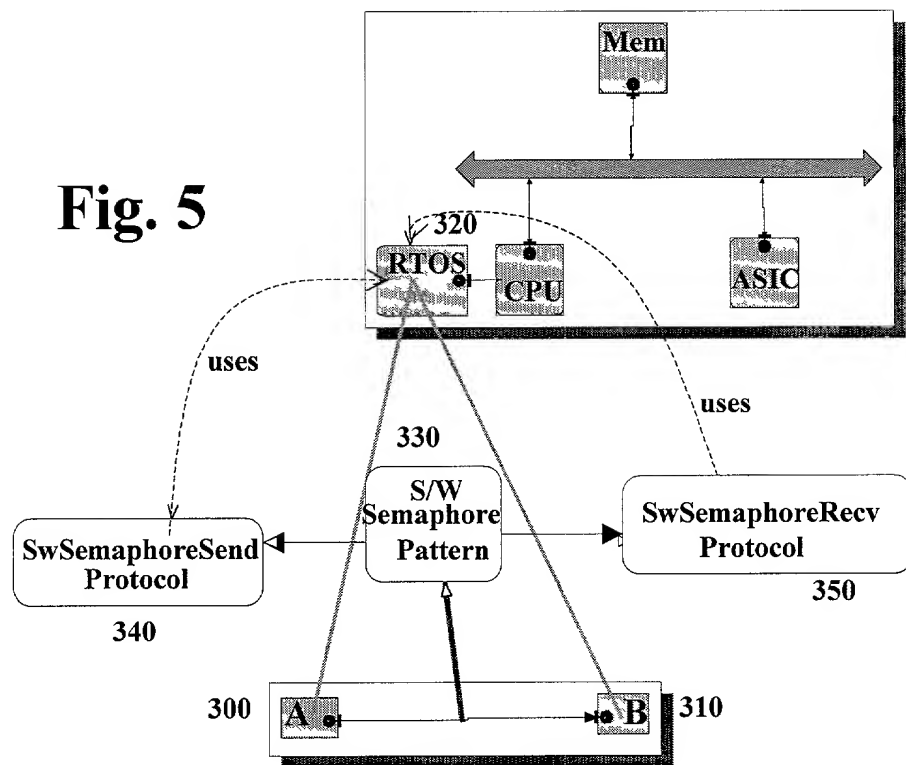


**Fig. 3**



**Fig. 4**

**Fig. 5**



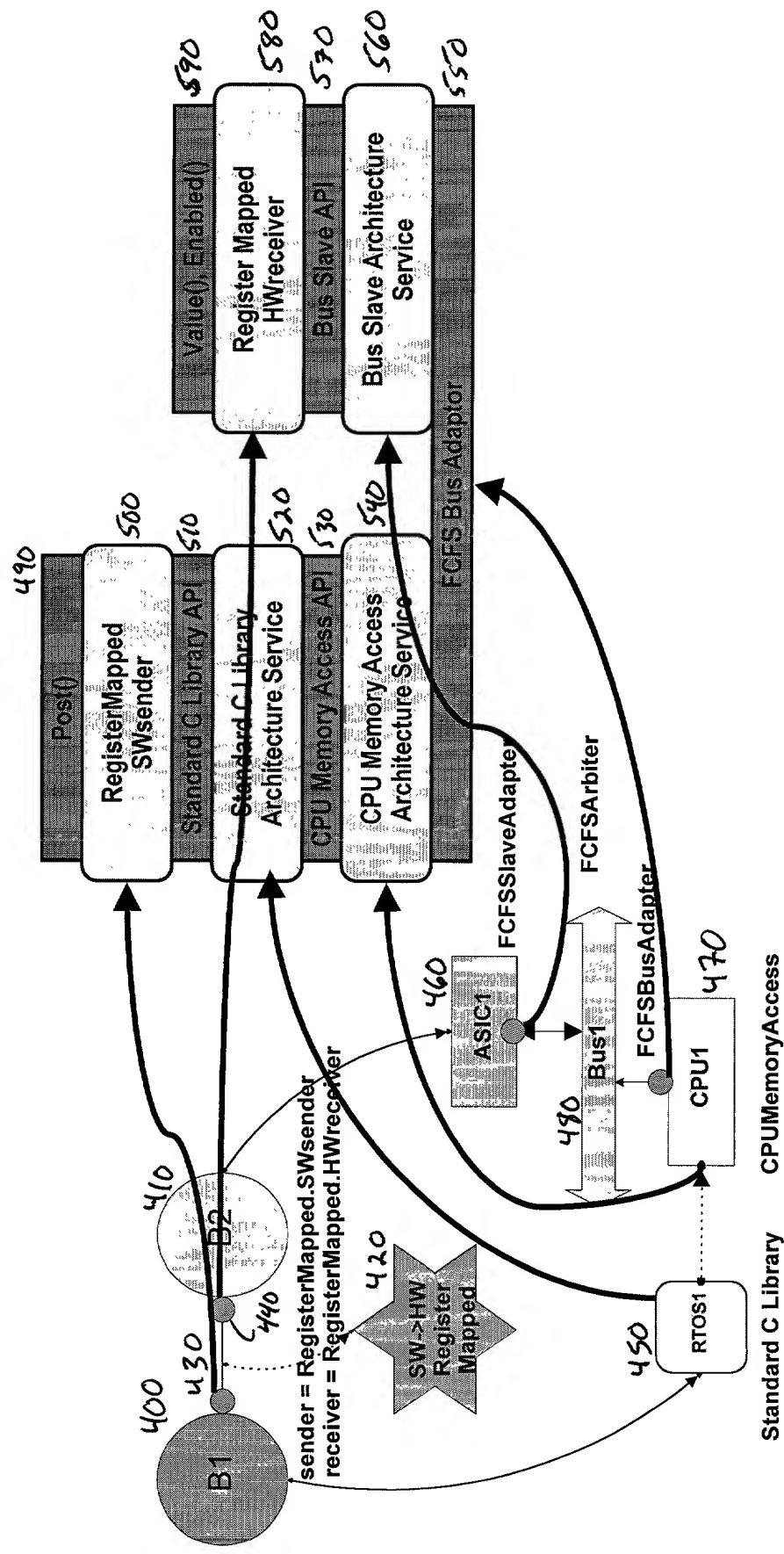
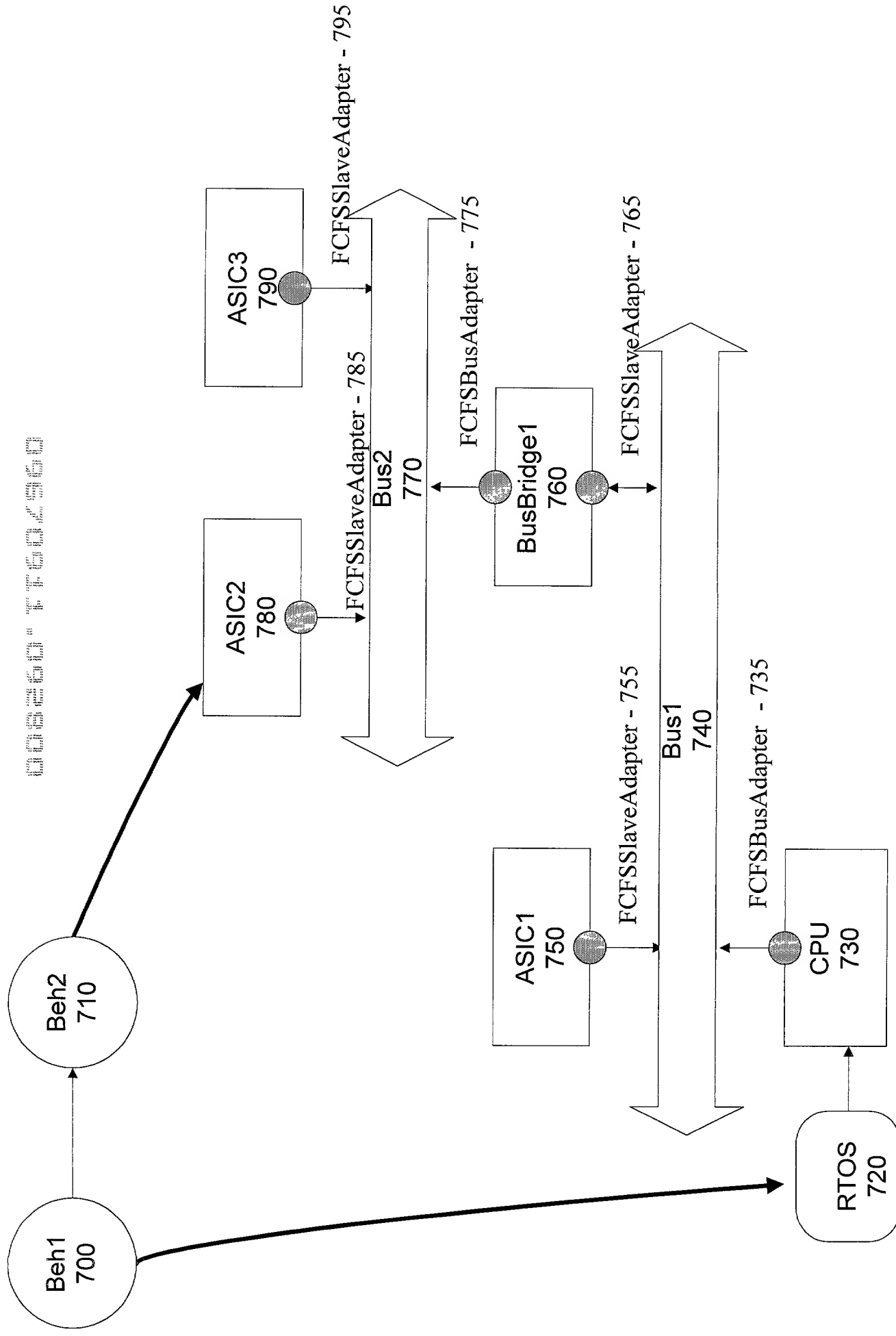


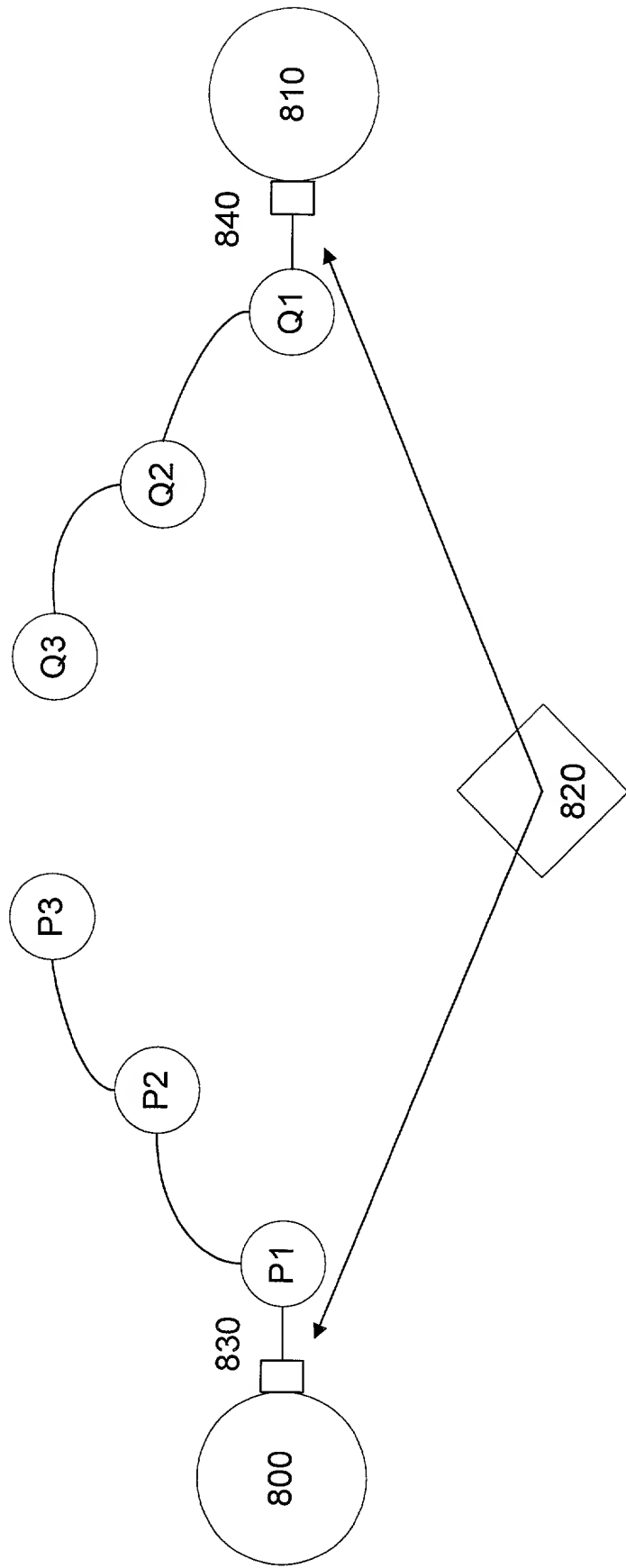
FIG. 6





**Fig. 8**

Fig. 9



## DECLARATION FOR PATENT APPLICATION

As a below named joint inventor, I hereby declare that:

My residence, post office address and citizenship are as stated below next to my name,

I believe I am an original and joint inventor of the subject matter which is claimed and for which a patent is sought on the invention entitled

**METHOD AND SYSTEM FOR PERFORMANCE LEVEL MODELING AND  
SIMULATION OF ELECTRONIC SYSTEMS HAVING BOTH HARDWARE AND  
SOFTWARE ELEMENTS**

the specification of which (check one)  X  is attached hereto or \_\_\_\_ was filed on \_\_\_\_\_, as Application No. \_\_\_\_\_ and was amended on \_\_\_\_ (if applicable).

I hereby state that I have reviewed and understand the contents of the above-identified specification, including the claims, as amended by any amendment referred to above.

I acknowledge the duty to disclose all information which is material to patentability as defined in 37 CFR § 1.56.

I hereby claim foreign priority benefits under 35 U.S.C. § 119(a)-(d) or § 365(b) of any foreign application(s) for patent or inventor's certificate, or § 365(a) of any PCT International application which designated at least one country other than the United States, listed below and have also identified below any foreign application for patent or inventor's certificate having a filing date before that of the application on which priority is claimed:

Prior Foreign Application(s)

Priority Claimed  
Yes No

| Number | Country | Day/Month/Year Filed |
|--------|---------|----------------------|
|--------|---------|----------------------|

| Number | Country | Day/Month/Year Filed |
|--------|---------|----------------------|
|--------|---------|----------------------|

I hereby claim the benefit under 35 U.S.C. § 119(e) of any United States provisional application(s) below.

Application No.: \_\_\_\_\_ Filing Date: \_\_\_\_\_

I hereby claim the benefit under 35 U.S.C. § 120 of any United States application(s), or § 365(c) of any PCT International application designating the United States, listed below and, insofar as the subject matter of each of the claims of this application is not disclosed in the prior United States application in the manner provided by the first paragraph of 35 U.S.C. § 112, I acknowledge the duty to disclose all information which is material to patentability as defined in 37 CFR § 1.56 which became available between the filing date of the prior application and the national or PCT international filing date of this application:

entitled: \_\_\_\_\_

| Application No. | Filing Date | Status: Patented, Pending, Abandoned |
|-----------------|-------------|--------------------------------------|
|-----------------|-------------|--------------------------------------|

| Application No. | Filing Date | Status: Patented, Pending, Abandoned |
|-----------------|-------------|--------------------------------------|
|-----------------|-------------|--------------------------------------|

Send correspondence to:

John W. Carpenter  
CROSBY, HEAFEY, ROACH & MAY  
P.O. Box 7936  
San Francisco, CA 94120-7936  
DID Ph.: 415-659-5927  
Ph.: 415-543-8700  
Fax: 415-391-8269

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment or both, under 18 U.S.C. § 1001 and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

Full name of joint and first inventor Sherry Solden

Inventor's signature \_\_\_\_\_

Date

Residence 4377 Miller Avenue, Palo Alto, CA 94306

Citizenship \_\_\_\_\_

Post Office Address Same

PATENT

Full name of joint and second inventor Edwin A. Harcourt

Inventor's signature \_\_\_\_\_

Date

Residence 4 Gemini Lane, Townsend, MA 01469

Citizenship \_\_\_\_\_

Post Office Address Same

Full name of joint and third inventor William W. La Rue, Jr.

Inventor's signature \_\_\_\_\_

Date

Residence 15296 Sherwood Road, Leawood, KS 66224

Citizenship \_\_\_\_\_

Post Office Address Same

Full name of joint and fourth inventor Douglas D. Dunlop

Inventor's signature \_\_\_\_\_

Date

Residence 11912 Hunting Ridge Court, Potomac, MD 20854

Citizenship \_\_\_\_\_

Post Office Address Same

Full name of joint and fifth inventor Christopher Hoover

Inventor's signature \_\_\_\_\_

Date

Residence 741 Pollard Road #4, Los Gatos, CA 95032

Citizenship \_\_\_\_\_

Post Office Address Same

**PATENT**

Full name of joint and sixth inventor Qizhang Chao

Inventor's signature \_\_\_\_\_

Date

Residence 2796 Louis Road, Palo Alto, CA 94303

Citizenship \_\_\_\_\_

Post Office Address Same

Full name of joint and seventh inventor Poonam Agrawal

Inventor's signature \_\_\_\_\_

Date

Residence 1063 Morse Avenue, #20-106, Sunnyvale, CA 94089

Citizenship \_\_\_\_\_

Post Office Address Same

Full name of joint and eighth inventor Aaron Beverly

Inventor's signature \_\_\_\_\_

Date

Residence 1063 Upper Ellen Road, Los Gatos, CA 95030

Citizenship \_\_\_\_\_

Post Office Address Same

Full name of joint and ninth inventor Massimiliano L. Chiodo

Inventor's signature \_\_\_\_\_

Date

Residence 10 Hillcrest Road, Berkeley, CA 94705

Citizenship \_\_\_\_\_

Post Office Address Same

**PATENT**

Full name of joint and tenth inventor Neeti K. Bhatnagar

Inventor's signature \_\_\_\_\_

Date

Residence 4301 Rennaissance Drive #127, San Jose, CA 95134

Citizenship \_\_\_\_\_

Post Office Address Same

Full name of joint and eleventh inventor Soumya Desai

Inventor's signature \_\_\_\_\_

Date

Residence Manhar Mahal, 4 Bakul Bagan Row, Flat #4C, Calcutta 700025, India

Citizenship \_\_\_\_\_

Post Office Address Same

Full name of joint and twelvth inventor Hungming Chou

Inventor's signature \_\_\_\_\_

Date

Residence 6822 Corte Munras, Pleasanton, CA 94566

Citizenship \_\_\_\_\_

Post Office Address Same

**PATENT**

Full name of joint and thirteenth inventor Michael D. Sholes

Inventor's signature \_\_\_\_\_

Date

Residence 1318 Edgewood Road, Redwood City, CA 94062

Citizenship \_\_\_\_\_

Post Office Address Same

Full name of joint and fourteenth inventor Ian Dennison

Inventor's signature \_\_\_\_\_

Date

Residence 555 River Oaks Parkway, San Jose, CA 95134

Citizenship \_\_\_\_\_

Post Office Address Same

Full name of joint and fifteenth inventor Sanjay Chakravarty

Inventor's signature \_\_\_\_\_

Date

Residence M36, Ramkrishna Vihar 29, I P Extension, Delhi 110092, India

Citizenship \_\_\_\_\_

Post Office Address Same

PATENT

Full name of joint and sixteenth inventor Eamonn O'Brien-Strain

Inventor's signature \_\_\_\_\_

Date

Residence 555 River Oaks Parkway, San Jose, CA 95134

Citizenship \_\_\_\_\_

Post Office Address Same

Full name of joint and seventeenth inventor Luciano Lavagno

Inventor's signature \_\_\_\_\_

Date

Residence 1571 Scenic Avenue, Berkeley, CA 94708

Citizenship \_\_\_\_\_

Post Office Address Same